Framework de jeu multijoueur en ligne de commande

Motivation

La nouvelle vague de recrutement de Hallebarde nécessite de sélectionner les meilleurs éléments parmi les candidats. Afin d'assurer la discrétion de ce programme de recrutement au sein de la population, il a été choisi mettre au point un serveur de jeu auquel peuvent se connecter les candidats avec tous types d'appareils. Il a également été décidé de concevoir le logiciel de manière modulaire, ce qui devrait nous permettre de larges gammes de scénarios et une potentielle réutilisation du programme de sélection pour d'autres objectifs.

Technologies

- Le langage de programmation Java permet la modularité souhaitée et assure la sécurité du logiciel quant à de potentielles attaques des services de renseignements, grâce notamment à sa gestion sécurisée de la mémoire. Il est de plus multiplateforme, ce qui évite les contraintes d'hébergement du serveur.
- L'absence de protocole spécifique au logiciel évite aux candidats d'avoir à télécharger un logiciel client pour interagir avec le serveur. La connexion se fait directement en mode texte par le protocole de transport TCP.
- L'intérêt du programme étant simplement d'évaluer les candidats face à divers scénarios, il n'est pas indispensable de disposer d'une interface complexe et le choix d'une IHM en ligne de commande suffit, tout en permettant de simplifier considérablement la couche réseau comme avec les avantages mentionnés plus haut.

Contenu de ce kit de Développement Logiciel

Le logiciel en lui-même étant limité au strict minimum, il est voué à être enrichi par des greffons (aussi appelés par leur dénomination anglo-saxonne "plugin"). Ces greffons interagissent avec le cœur de l'application via une interface applicative de programmation (API, de son acronyme anglo-saxon). Le présent kit contient les outils nécessaires au développement de greffons.

Répertoires

- `javadoc/` continent la documentation de l'API de développement de greffon, elle peut être ouverte dans un navigateur.
- `example-plugin/` contient un projet gradle pouvant servir de base au développement d'un greffon, avec par conséquent les librairies nécessaires.

Utilisation du kit

Vous devez disposer d'une installation de Java fonctionnelle, en version 17. Vous pouvez ouvrir le projet `example-plugin` dans votre IDE préféré. Dans le cas d'Eclipse, il vous faudra configurer le projet en exécutant la commande suivante depuis la racine du projet :

Linux & macOS

```
./gradlew eclipse
```

Windows

```
gradlew.bat eclipse
```

Au cours de votre développement, vous pourrez exécuter le logiciel avec votre greffon via la commande suivante :

Linux & macOS

```
./gradlew runServerWithPlugins
```

Les fichiers de configuration du serveur se trouvent dans son répertoire de travail, nommé `run`.

Windows

```
gradlew.bat runServerWithPlugins
```

Vous pouvez compiler votre plugin pour en obtenir le jar avec la commande suivante :

Linux & macOS

```
./gradlew build
```

Windows

```
gradlew.bat build
```

Le jar se trouvera ensuite dans le répertoire `build/libs`.

Concepts de programmation

Structure d'un greffon

Chaque greffon est constitué d'un fichier jar. Le point d'entrée d'un greffon est une classe implémentant l'interface `Plugin`, et renseigné dans le fichier `plugin.json` à la racine du jar. Ce fichier contient les métadonnées du greffon :

```
{
  "pluginId": "example",
  "mainClass": "com.example.plugin.ExamplePlugin",
  "version": "1.0.0"
}
```

Chaque greffon est identifié par un identifiant qui doit être unique (ici, `example`).

Événements

La manière principale pour un greffon d'interagir avec le cœur de l'application est en interceptant des événements. Il est possible de renseigner une méthode pouvant traiter un événement auprès du logiciel via la méthode `Game#registerEventHandler(Object)`. La méthode devra être publique, statique et avoir pour unique paramètre un object héritant de la class `Event`. La liste complète des évènement se trouve dans le package `org.hallebarde.recrutement.api.events`.

Commandes

Les commandes que peuvent exécuter les utilisateurs sont implémentée via l'interface `Command`. Elles doivent être renseignées auprès du serveur via la méthode `Game#registerCommand(String, Command, String)`. L'utilisateur invoque une command à partir de son prefix, précédé d'un slash. Par example, pour invoquer la commande d'aide, dont le prefix est `help`, un joueur enverra la ligne `/help`. La commande help permet d'obtenir une liste des commandes installées. Toute entrée utilisateur qui n'est pas une commande est envoyée dans le chat, visible par tous les joueurs connectés.

Salles

Le monde du serveur de jeu est divisée en salles reliées les unes autres. Elles sont configurées dans le dossier du monde et contiennent chacune un identifiant, une description, la liste des salles voisines accessibles, la liste des objets présents dans la salle et la liste des interactions possible dans cette salle.

Activités

Les activités sont des contextes particuliers qui peuvent être ajoutés par les greffons. Lorsqu'un joueur est engagé dans une activité, cette dernière intercepte toutes les entrées du joueur, et ce dernier ne peut donc plus interagir avec le chat ou exécuter des commandes. Les greffons peuvent rajouter des activités en étendant la class `Activity`. Le cycle de vie d'une activité peut être contrôlé via les méthodes `Game#startActivityNow(Activity)`, `Game#startActivityWhenPossible(Activity)` et `Game#stopActivity(Activity)`. Le serveur ne contient aucune activité par défaut.

Interactions

Les interactions permettent au joueur d'interagir avec son environnement. Une interaction est fixée dans une salle, à la différence d'un item. Un joueur peut interagir avec la commande `/interact`. Une conversation avec un personnage non-joueur peut par exemple être implémenté avec une interaction. Les greffons peuvent rajouter des interactions en implémentant l'interface `Interaction` et en informant le serveur de l'existence de l'activité via la `Registry` d'interaction, dont l'instance peut être obtenue via `Game#getRoomInteractionRegistry()`. Le serveur ne contient aucune interaction par défaut.

Items

Les items sont similaires aux interactions, mais peuvent être récupérés et transportés par le joueur via l'inventaire. Un joueur peut récupérer un item dans une salle via la commande `/pickup`, le reposer via la commande `/drop`, ou l'utiliser via la commande `/use`. Un joueur peut voir les items dans son inventaire via la commande `/inventory`. Les greffons peuvent rajouter des items en implémentant l'interface `Item` et en informant le serveur de l'existence de l'item via la `Registry` d'items, dont l'instance peut être obtenue via `Game#getRoomItemRegistry()`. Le serveur ne contient aucun item par défaut.